ACSLS Backup

Bacula Systems Documentation

Contents

1 Overview **3**
- 1.1 Executive summary . . . 3
- 1.2 Features Summary . . . 3

2 Theory of Operations **3**
- 2.1 ACSLS Advantages . . . 3
- 2.2 General ACSLS Operation . . . 4

3 Bacula Enterprise Integration **4**
- 3.1 ACSLS Changer . . . 4

4 Installation **5**
- 4.1 Installation of the Plugin . . . 5

5 Configuration **5**
- 5.1 Changer Configuration . . . 5
- 5.2 Storage Daemon Configuration . . . 6
- 5.3 SSI Component Configuration . . . 8
- 5.4 DRIVEID mapping . . . 9

6 Operations **10**
- 6.1 Testing ACSLS Integration . . . 10
- 6.2 Other commands . . . 11
- 6.3 Best practice . . . 13

7 Other **15**
- 7.1 Supported Platforms . . . 15
- 7.2 Limitations . . . 16

Contents

- *Overview*
- *Theory of Operations*
- *Bacula Enterprise Integration*
- *Installation*
- *Configuration*
- *Operations*
- *Other*

1 Overview

1.1 Executive summary

This document is intended to provide insight into the considerations and processes required to integrate **Oracle ACSLS** with **Bacula Enterprise**.

1.2 Features Summary

- Provide all tape library management operations required by **Bacula Enterprise**.
- Support named user access to ACSLM if required.
- Support tape drive and volume locking in shared **ACSLS** environment.
- Support lock query and management.
- Static tape drive location mapping.
- Dynamic volume location mapping.

2 Theory of Operations

2.1 ACSLS Advantages

ACSLS offers many advantages when managing a tape library environment:

- Processes multiple requests in parallel and optimizes use of large library complexes.
- Avoids delays caused by pass-thru between robots.
- Automatically recovers and retries requests that fail.
- Allows multiple clients to share a library.
- Simplifies support of new libraries and library features.
- Provides a choice of interfaces.
- Presents logical libraries through a SCSI media changer interface over fibre channel.
- Changes library configurations while libraries remain online.
- Manages all libraries at a customer site from **ACSLS**.
- Provides a high availability option.

A major feature of **Oracle ACSLS** is the ability to manage any combination of tape libraries. This provides access to the latest ACS technology and to applications across libraries.

2.2 General ACSLS Operation

The **ACSLS** acts as a library management system. **ACSLS** manages the physical aspects of tape cartridge storage and retrieval through a system administrator interface and a programmatic interface. These real–time interfaces control and monitor tape libraries, including access control. **ACSLS** does not have access to the data path through which information is read from or written to tape cartridges. **ACSLS** operates only on the control path through which libraries are managed.

The **ACSLS** consists of the following:

- One or more tape libraries connected with pass-thru ports.
- Library(ies) that contains tape cartridges uniquely identified by an external label with a volume serial number (VOLSER) and a media domain and type.
- A set of tape drives (sometimes referred to as transports) in the library or libraries.
- A set of cartridge access ports (CAPs) attached to the library which allow cartridges to be physically entered or removed from the library.
- Robotics that physically move cartridges between library storage cells, cartridge drives, CAPs, and pass-thru ports.

Library requests originate from client applications running on a host system. These requests and messages then move across a network connecting client systems and the ACSLS. An example of a client application is Bacula Enterprise backup and restore software.

3 Bacula Enterprise Integration

Bacula Enterprise ACSLS allows management of ACS using ACSAPI for all required operations. The integration consist of three main components:

- `acsls-changer` - The application which interacts between Bacula Storage Daemon and makes requests to the tape library system and ACSLM system.
- `ssi` - The SSI component of **ACSLS** provided by **Bacula Enterprise** which could be used as a replacement for the standard SSI component distributed by **ACSLS** software vendor.
- `ssi_el` - The SSI Event Logger component required for saving event and trace logs generated by the SSI component provided by **Bacula Enterprise**.

3.1 ACSLS Changer

Bacula Enterprise's `acsls-changer` is a direct replacement for the `mtx-changer` script which manages tape libraries connected to ACSLM. You can use its services directly by manually inputing required commands and parameters or indirectly by Bacula Storage Daemon configuration and standard `bconsole` commands.

4 Installation

The first step is to install and configure the Bacula Storage Daemon on the backup machine which will manage tape libraries using ACSLM and **ACSLS** integration. Next, install the corresponding ACSLS Plugin package on this Storage Daemon.

4.1 Installation of the Plugin

On the Bacula Storage Daemon that you want to connect to your ACSLM, extend the repository file for your package manager to contain a section for the plugin. For example in `Red Hat/CentOS`, `/etc/yum.repos.d/bacula.repo`:

```
[Bacula]
name=Bacula Enterprise
baseurl=https://www.baculasystems.com/dl/@customer-string@/rpms/bin/@version@/rhel7-64/
enabled=1
protect=0
gpgcheck=0

[BaculaACSLS]
name=Bacula Enterprise ACSLS Plugin
baseurl=https://www.baculasystems.com/dl/@customer-string@/rpms/acsls/@version@/rhel7-64/
enabled=1
protect=0
gpgcheck=0
```

Then perform a yum update, and after that the package *bacula-enterprise-acsls* can be installed with `yum install`. If you prefer to manually install the packages, you can also download them from your download area, and use the low level package manager tool (`rpm`) to do the plugin installation.

5 Configuration

After installation of the **Bacula Enterprise ACSLS**, it should be configured to meet your backup environment requirements, including the required `acsls-changer` configuration and optional `SSI` component configuration.

5.1 Changer Configuration

The plugin is configured using a dedicated configuration file `acsls-changer.conf` located in the */opt/bacula/etc* directory, and a proper Bacula Storage Daemon resource configuration.

Changer Config File

The `acsls-changer.conf` file is a simple `key=value` configuration file with the following parameters:

- `ACSAPI_PACKET_VERSION=<nr>` specifies the ACSLM supported packet version (default 4). It should match the corresponding SSI parameter. This parameter is optional. If not set, the default will be used.
- `ACSAPI_SSI_SOCKET=<port>` specifies the socket for SSI interprocess communication (default 50004). This parameter is optional. If not set, the default will be used.

- VOLDBT=<path/file> specifies the acsls-changer working file which stores the volumes-to-slot mappings. The path/file value should be set to /opt/bacula/working/acsls-changer.dbt. This parameter is optional. If not set, a file named acsls-changer.dbt in the current working directory will be used.
- USERID specifies the default ACSAPI user. This parameter is optional. If not set, no user information will be sent to ACSLM.
- LOCKING=<yes|no> specifies if acsls-changer should use the resource (Drive and Volume) locking mechanism. This parameter is optional. If not set, no resource locking will be used. See: *Resource locking*.
- DRIVE<N>=<acs:lsm:panel:drive> specifies a static mapping between a Bacula parameter configured in a Bacula Storage Daemon resource configuration and ACSLS DRIVEID. Every tape drive defined in a bacula-sd.conf resource which is managed by **ACSLS** should be defined as an appropriate parameter in the config file. This parameter is required and should be defined at least once. See: *DRIVEID mapping*.
- TIMEOUT=<secs> specifies how long in seconds the acsls-changer should wait for a request to complete (default is 600 seconds). A value of zero means no timeout (infinite wait). The TIMEOUT value should not exceed the Maximum Changer Wait time configured in the Bacula Storage Daemon bacula-sd.conf file. See: *Request timeout*.

Here is an acsls-changer.conf file example:

```
ACSAPI_PACKET_VERSION = 4
ACSAPI_SSI_SOCKET = 50004
USERID = root
LOCKING = yes
VOLDBT = /opt/bacula/working/acsls-changer.dbt
DRIVE0 = 1:2:3:4
DRIVE1 = 1:2:3:5
```

5.2 Storage Daemon Configuration

To use an **ACSLS** storage in **Bacula** you should configure Autochanger/Device resources in the same way as for a standard tape library or tape autochanger managed by the mtx utility. The only exception is an updated Changer Command as in this example:

```
Changer Command = "/opt/bacula/bin/acsls-changer %o %d %S"
```

Where:

- **%o** is a changer command (load, unload, etc)
- **%d** is a index number from the Drive Index parameter of a Device resource
- **%S** is a slot number (started from 1)

Here is an example and resource configuration to use with **ACSLS**.

```
#
# An ACSLS system with two drives
#
Autochanger {
   Name = SL8500-ACSLS
   Device = IBMLTO7-1
   Device = IBMLTO7-2
   # %o=command, %d=drive-index, %S=slot(base1)
   Changer Command = "/opt/bacula/bin/acsls-changer %o %d %S"
```

(continues on next page)

(continued from previous page)

```
   Changer Device = /dev/null
}
Device {
   Name = IBMLTO7-1
   DriveIndex = 0
   DeviceType = Tape
   MediaType = LTO7
   ArchiveDevice = /dev/tape/by-id/scsi-3500104f000899ece-nst
   AutomaticMount = no
   AlwaysOpen = no
   RemovableMedia = yes
   RandomAccess = no
   AutoChanger = yes
   OfflineOnUnmount = yes
}
Device {
   Name = IBMLTO7-2
   DriveIndex = 1
   DeviceType = Tape
   MediaType = LTO7
   ArchiveDevice = /dev/tape/by-id/scsi-3500104f000899ed1-nst
   AutomaticMount = no
   AlwaysOpen = no
   RemovableMedia = yes
   RandomAccess = no
   AutoChanger = yes
   OfflineOnUnmount = yes
}
```

Note: On some systems and tape libraries managed by **ACSLS** it might be required to ummount the tape drive by ejecting the tape after use (unmount). This can help **ACSLS** to allow the tape robot to grab the tape from the drive. Due to the above, the following parameter should be defined in the Bacula Storage Daemon configuration for every drive device:

```
OfflineOnUnmount = yes
```

This configuration setting is present in the example above.

5.3 SSI Component Configuration

You may use the SSI component provided by your **ACSLS** vendor or use the SSI component from the **Bacula Enterprise ACSLS** plugin. To use SSI component from the **Bacula Enterprise ACSLS** plugin you should prepare an SSI config file as described in the next section.

SSI Config file

The acsls-ssi.conf file is a simple key=value configuration with the following parameters:

- CSI_HOSTNAME=<address> defines the CSI component location address. Acceptable values are an IP address or a fully qualified domain name for the host. This parameter is required.
- CSI_TCP_RPCSERVICE=<TRUE/FALSE> is used to define whether the CSI operates as a TCP RPC Server (default TRUE). This variable must be set to TRUE for the firewall-secure CSC. The firewall-secure **ACSLS** application packets are all sent using the TCP network transport. This parameter is optional. If not set, the default will be used.
- CSI_UDP_RPCSERVICE=<TRUE/FALSE> is used to define whether the CSI operates as a UDP RPC server (default TRUE). This variable must be set to FALSE for the firewall-secure CSC. The firewall-secure **ACSLS** application packets are all sent using the TCP network transport. The CSI can operate as a TCP and a UDP server simultaneously. This parameter is optional. If not set, the default will be used.
- CSI_CONNECT_AGETIME=<nr> defines the value of the maximum age of pending requests in the CSI's request queue. This variable is expressed as an integer representing a number of seconds. A value of 172800 indicates two days which is the default. Messages older than this value are removed from the queue and the CSI sends an entry to the Event Log when this happens. This parameter is optional. If not set, the default will be used.
- CSI_RETRY_TIMEOUT=<nr> defines the minimum amount of time, in seconds, that the CSI will wait between attempts to establish a network connection (default 4). This parameter is optional. If not set, the default will be used.
- CSI_RETRY_TRIES=<nr> defines the number of attempts the CSI will make to transmit a message (default 5). Pending messages are discarded if a connection cannot be established within the defined number of tries. The CSI_RETRY_TIMEOUT and CSI_RETRY_TRIES together determine the minimum total time the CSI will attempt to send a message. This parameter is optional. If not set, the default will be used.
- ACSAPI_SSI_SOCKET=<port> is the socket port number (default 50004) on which the SSI component listens to incoming requests from acsls-changer. This parameter should be the same as ACSAPI_SSI_SOCKET defined in the acsls-changer.conf file. This parameter is optional. If not set, the default will be used.

Here is an acsls-ssi.conf file example:

```
CSI_HOSTNAME=10.10.10.1
ACSAPI_SSI_SOCKET=50009
CSI_UDP_RPCSERVICE=FALSE
```


5.4 DRIVEID mapping

To configure drive mapping in the `acsls-changer.conf` file you should use tape drive serial number matching. To obtain the serial numbers from the **ACSLS** managed tape library drives, execute the following commands:

```
root@saxo:~# su - acsss
bash-4.1# cd bin
bash-4.1# ./cmd_proc
-------Oracle ACSLS (Automated Cartridge System Library Software) 8.4.0-3------

ACSSA> display drive * -f serial_num
2018-03-12 05:07:17        Display Drive
Acs  Lsm  Panel  Drive  Serial_num
2    0    1      11     1013001383
2    0    1      15     1013001384
```

In the example above you can see two drives and their corresponding serial numbers. Then check the serial number of all drives attached to your Bacula Enterprise Storage server using the `tapeinfo` command.

To use this command you will need SCSI generic devices (eg: /dev/sgX) for all of your tape drives:

```
# lsscsi -g
(...)
[8:0:7:0]    tape    IBM      ULTRIUM-TD7      G9Q2  /dev/st0   /dev/sg6
[8:0:8:0]    tape    IBM      ULTRIUM-TD7      G9Q2  /dev/st2   /dev/sg8
```

Then, for each SCSI generic device (/dev/sgX) listed for your tape drives, get its serial number. For example:

```
# tapeinfo -f /dev/sg6
Product Type: Tape Drive
(...)
SerialNumber: '1013001383'
(...)

# tapeinfo -f /dev/sg8
Product Type: Tape Drive
(...)
SerialNumber: '1013001384'
(...)
```

Then match all available tape drive serial numbers in your storage server to the serial numbers from **ACSLS**. You should use `Acs`, `Lsm`, `Panel`, `Drive` to define a drive mapping in the `acsls-changer.conf` file as in the above example.

The `acsls-changer.conf` drive mapping for the above example would be configured as follows:

```
(...)
DRIVE0 = 2:0:1:11
DRIVE1 = 2:0:1:15
```

and the corresponding resources in `bacula-sd.conf`:


```
Device {
   Name = TapeDrive0
   Drive Index = 0
   Archive Device = /dev/nst0
(...)
}
Device {
   Name = TapeDrive1
   Drive Index = 1
   Archive Device = /dev/nst2
(...)
}
```

6 Operations

This chapter describes runtime operations for **ACSLS** management and testing with **Bacula Enterprise** integration.

After editing the `acsls-changer.conf` and `acsls-ssi.conf` files, the `/opt/bacula/scripts/acsls-start.sh` must be run to start the Bacula ssi component (replacement for the standard one from ACSLS vendor), and `ssi_el` (event logger) processes.

```
# /opt/bacula/scripts/acsls-start.sh
```

6.1 Testing ACSLS Integration

Before attempting to use the **ACSLS** integration with **Bacula**, it is preferable to manually test that the integration works. To do so, we suggest you perform the following steps:

listall command

This command should list all defined drives and available volumes. The command can take some time to complete depending on the number of available volumes in ACSLM. If an error is printed instead of a list please contact Bacula Systems support.

```
# /opt/bacula/bin/acsls-changer listall
D:0:E
D:1:F:2:F70170
S:1:F:F70168
S:2:E
S:3:F:F71070
S:4:F:F71071
S:5:F:F71072
S:6:F:F71073
S:7:F:F71074
S:8:F:F71075
S:9:F:F71076
S:10:F:F71077
S:11:F:F71078
```


load command

This command should load the volume from the specified slot into the specified drive. If an error is printed instead of a 'done', the error message may provide enough information for easy troubleshooting. If not, please contact Bacula Systems support.

```
# /opt/bacula/bin/acsls-changer load 0 5
Loading Volume F71072 into drive 0 ... done
```

loaded command

This command allows you to check if a volume is loaded into a specified drive. The number printed is a volume slot loaded into the drive. If the value printed is zero, then no volume is loaded in the specified drive. As above, if an error is printed instead of a slot number, you can attempt to resolve the problem printed in the error message or ask for support.

```
# /opt/bacula/bin/acsls-changer loaded 0
5
```

unload command

This command allows you to unload the volume from the selected drive. If an error is printed instead of 'done' you can attempt to resolve the problem printed in the error message or ask for support.

```
# /opt/bacula/bin/acsls-changer unload 1
Unloading Volume F70170 from drive 1 ... done
```

6.2 Other commands

The main purpose for the `acsls-changer` command is to allow a Bacula Enterprise Storage Daemon to manage ACSLS tape libraries, but the command was also designed to allow management operations to be performed manually.

```
# /opt/bacula/bin/acsls-changer -?
Copyright (C) 2000-2017 Kern Sibbald.

Version: 9.0.6 (20 November 2017)

Usage: acsls-changer [options] <command> [<drive-index> <slot> <volumename>]
-d <nn>            set debug level to <nn>
-c <file>          acsls-changer config file
-b <file>          acsls-changer volumes database file
-?                 print this message
```

The utility supports the following commands:

- `list` - prints the list of all available volumes and slots.
- `listall` - prints the list of all defined drives and all available volumes and slots. See *listall command*.
- `load <driveindex> <slot> [volume]` - loads a volume specified by a `slot` or `volume` into a selected tape drive specified by a `<driveindex>`. If you specify a `<volume>` parameter then the slot number will be ignored. You can use the `<volume>` parameter on manual operations only. This kind of operation is not supported by Bacula Storage Daemon. See: *load command*.

- `unload <driveindex>` - unloads a volume from a tape drive specified by a `<driveindex>`. See: *unload command*.
- `slots` - prints a number of all available slots including free slots.
- `loaded <driveindex>` - prints a volume slot number loaded into a drive. See: *loaded command*.
- `showdrvlock <driveindex>` - prints information about a tape drive status and lock. See: *showdrvlock command*.
- `showvollock <volume>` - prints information about a volume status and lock. See: *showvollock command*.
- `cleardrvlock <driveindex>` - clears a lock assigned to the specified drive. See: *cleardrvlock command*.
- `clearvollock <volume>` - clears a lock assigned to the specified volume name. See: *clearvollock command*.
- `showvollockall` - prints lock information about all available volumes. See: *showvollockall command*.

showdrvlock command

To print the lock information about a specific tape drive, use the following command example:

```
# /opt/bacula/bin/acsls-changer showdrvlock 0
ACS_QUERY_LOCK_DRV_RESPONSE: STATUS_SUCCESS (0)

DRV: [2:0:1:15]
    lock:14509 userid: regress
    duration:67 pending: 0
    status STATUS_DRIVE_IN_USE (29)
```

When no lock is assigned to the specified tape drive no lock information will be printed.

showvollock command

To print the lock information about a specific volume, use the following command example:

```
# /opt/bacula/bin/acsls-changer showvollock F71072
ACS_QUERY_LOCK_VOL_RESPONSE: STATUS_SUCCESS (0)

VOL: F71072
    lockid:14509 userid: regress
    duration:92 pending: 0
    status STATUS_VOLUME_IN_USE (99)
```

When no lock is assigned to the specified volume no lock information will be printed.

cleardrvlock command

To clear the lock assigned to a tape drive, use the following command example:

```
# /opt/bacula/bin/acsls-changer cleardrvlock 0
ACS_CLEAR_LOCK_DRV_RESPONSE: STATUS_SUCCESS (0)

DRV: [2:0:1:15]
    status STATUS_SUCCESS (0)
```


When no lock is assigned then `STATUS_DRIVE_AVAILABLE` will be returned.

clearvollock command

To clear the lock assigned to a volume, use the following command example:

```
# /opt/bacula/bin/acsls-changer clearvollock F71072
ACS_CLEAR_LOCK_VOL_RESPONSE: STATUS_SUCCESS (0)

VOL: F71072
    status STATUS_SUCCESS (0)
```

When no lock is assigned then `STATUS_VOLUME_AVAILABLE` will be returned.

showvollockall command

To print lock information for all available volumes use the following command example:

```
# /opt/bacula/bin/acsls-changer showvollockall
ACS_QUERY_LOCK_VOL_RESPONSE: STATUS_SUCCESS (0) Vols: 2

VOL: F71072
    lockid:27968 userid: regress
    duration:2553 pending: 0
    status STATUS_VOLUME_IN_USE (99)

VOL: F71073
    lockid:6903 userid: regress
    duration:2424 pending: 0
    status STATUS_VOLUME_IN_USE (99)
```

The number of volumes which have locks assigned is printed as **Vols:<nr>**.
When no volumes with assigned locks are found then the number of volumes will be zero.

6.3 Best practice

Label Volumes

To use tape volumes in Bacula Enterprise they must first be labeled. It is recommended to use the command `label barcodes` which will automatically assign **ACSLS** Volume names to the **Bacula** volumes.

Note: It is not recommended (but technically possible) to use a different **Bacula** volume name for an **ACSLS** Volume.

Access Control

To use the **ACSLS** Access Control feature you should enable it in your ACSLM software (verify *ACSLS User Guide* for this) and define a `USERID` parameter in `acsls-changer.conf` file. Without it your Access Control won't work. With **ACSLS** Access Control you can define fine-grained access rules to your resources including allowed commands and volumes (with *Volume Access Control*). The following commands are used by `acsls-changer` and cannot be disabled:

- `mount`
- `dismount`
- `dismount_force`
- `query`
- `lock` - when resource locking is enabled, see: *Resource locking*
- `query_lock` - when resource locking is enabled
- `clear_lock` - when resource locking is enabled

With *ACSLS Volume Access Control*, you can limit the volume pool available for **Bacula**. Consult your *ACSLS User Guide* for information on setting up volume pool limitations.

Resource locking

You can use the ACSLS resource locking mechanism to protect tape drive and volume resources when used by a Bacula Enterprise Storage Daemon. To use this mechanism you need to define a `USERID=<userid>` and `LOCKING=Yes` parameters in the `acsls-changer.conf` file (See: *Changer Config File*). If it is not strictly required to avoid resource locking in your environment, you can disable resource locking by simply removing the `LOCKING` parameter from the config file or set it to `No`.

Request timeout

Some **ACSLS** requests may take longer than expected due to busy tape libraries or other hardware issues. If you get the following error code:

```
# ./acsls-changer unload 1 2 F70170
Unloading Volume F70170 from drive 1 ... code 296 (296)
```

it means a timeout has occured and you should verify that this is the problem. It may be necessary to modify the `TIMEOUT` value in the `acsls-changer.conf` file (See: *Changer Config File*) and `Maximum Changer Wait` in the Bacula Storage Daemon `bacula-sd.conf` file.

You can get different timeout messages during mount ("*Timeout on volume available status!*") or unmount ("*Timeout on verify volume status!*") operations. These messages mean that Bacula is unable to confirm the expected volume status during these operations before a timeout. If you get these timeout messages you should verify a real volume status in ACSLS. If required, modify the `TIMEOUT` value in the `acsls-changer.conf` file (See: *Changer Config File*) and `Maximum Changer Wait` in the Bacula Storage Daemon `bacula-sd.conf` file.

Stalled lockid

During operations, if you get an error message about a stalled lockid, then you will need to manuallly clear the lock outside `acsls-changer` (i.e. using a `clear lock` of `cmd_proc` application).

```
Error: Stalled lockid found (nnn). Make sure that the system is not shared (...)
```

This makes the `acsls-changer` out of sync. In this case you should verify what caused a lockid to disappear from the **ACSLS** system and remedy this. One typical cause of this can be a shared `acsls-changer` setup where one installation creates resource locks and other does not. This kind of setup is unsupported.
When you've removed the root cause of this error it could be required to sync the `acsls-changer` with **ACSLS** system again otherwise you will get the same error again. To do this, you have to first stop the Bacula Storage Daemon then remove the first line from VOLDB file (`/opt/bacula/working/acsls-changer.dbt`) which should contain the following value:

```
LOCKID:nnn
```

If the first line of `/opt/bacula/working/acsls-changer.dbt` file does not contain a LOCKID word then you should not modify this file.

Trace file

You can enable debug trace file generation using the following `acsls-changer` command parameters: `-d<nn>`, `-dt` and `-T`, where `nn` is a required debug level, i.e.

```
Changer Command = "/opt/bacula/bin/acsls-changer -d100 -dt -T %o %d %S"
```

Then the debug trace file will be generated at: `/opt/bacula/working/acsls-changer.trace` The option `-T` enables trace file generation and option `-dt` add timestamps to every debug log.

7 Other

7.1 Supported Platforms

The Bacula Enterprise ACSLS Plugin is supported on the following platforms:

- `Red Hat/CentOS` version 6 and 7
- `SLES` version 11 and 12

7.2 Limitations

- **ACSLS** Media Management feature is not supported. You should use a standard **Bacula** `Media Type` directive to manage different media.
  This limitation may be removed in the future.
- **ACSLS** dynamic drive allocation feature is not supported. **Bacula** uses a static drive mapping defined in `acsls-changer.conf` config file.
  This limitation may be removed in the future.
- **ACSLS** Scratch Pools management feature is not supported. **Bacula** uses its own Scratch Pool management.
  This limitation may be removed in the future.
- **Bacula** `mtx-changer transfer` command is not supported as there is no such functionality in **ACSLS**.
- The `restart` command has limitations with plugins, as it initiates the Job from scratch rather than continuing it. Bacula determines whether a Job is restarted or continued, but using the `restart` command will result in a new Job.

